

Database Processing

CS 451 / 551



Suyash Gupta

Assistant Professor

Distopia Labs and ONRG

Dept. of Computer Science

(E) suyash@uoregon.edu

(W) [gupta-suyash.github.io](https://github.com/gupta-suyash)



UNIVERSITY OF
OREGON

Welcome!

Course Webpage:

<https://gupta-suyash.github.io/cs451-fall25.html>

Course Canvas:

<https://canvas.uoregon.edu/courses/279438>

Instructors

- **Suyash Gupta**
 - Office Hours: Tuesday, 10-11am, Deschutes 334
- **Ethan Childs (TA)**
 - Office Hours: Monday /Wednesday, 12-1pm, Deschutes 227

TextBooks

- **Main Course Book:**
 - Database System Concepts (7/e by Silberschatz).
- **Optional Reading:**
 - Database Management Systems (3/e by Ramakrishnan).

Grading Scheme

- 30% - Assignments
- 10% - Presentation
- 10% - Quiz 1
- 10% - Quiz 2
- 20% - Final
- Undergraduate Students
 - 20% - Participation (Attendance + Class Discussions)
- Graduate Students
 - 10% - Participation (Attendance + Class Discussions)
 - 10% - Term Paper

Grading Rubric

- Rubric
 - 97% - A+
 - 92% - A
 - 87% - A-
 - 82% - B+
 - 77% - B
 - 72% - B-
 - 67% - C+
 - 62% - C

Assignments

- Assignments will be completed in groups of at most 4.
- Every student in the group needs to state their contribution.

Assignments

- Total **three assignments**: each assignment carries a weight of **10%**.
 - Total Assignment weight: **30%**.
- **Auto-grading**:
 - We will give you scripts to test your code.
 - We will have hidden test cases to evaluate your code.

Assignments

- **You have total 3 late days.** You can use them all in one assignment, or one late day per assignment.
 - No additional late days will be given.
- **Deadline is strict. It will be always at 11:59pm.**

Assignment Schedule

- **Assignment 1:**
 - Release Date: October 7th 2025.
 - Due Date: October 28th 2025.
- **Assignment 2:**
 - Release Date: October 29th 2025.
 - Due Date: November 13th 2025.
- **Assignment 3:**
 - Release Date: November 14th 2025.
 - Due Date: November 30th 2025.

Presentation

- **Presentation:**
 - In the dead week (last week) before final exams.
 - 15-20min each group.
 - Each member of the group will present.
 - Each member of the group will be asked questions on the part they completed.
 - More details later.

Term Paper for Graduate Students

- **Goal – Exploring Beyond Intro.**
- Selecting one topic and researching **1 state-of-the-art paper.**
- Giving your opinions and thoughts in a **4-page conference style paper.**
- More details later.

- **Due Date: Dec 1st, 2025 at 11:59pm.**

Plagiarism

- Please don't plagiarize.
- Just don't!
- We will apply the harshest UO policy available to us to deal with plagiarism!
- More details on the webpage.

Missed Attendance Policy

- You can miss **at most 2 classes** without getting penalized.

Database Processing

CS 451 / 551

Lecture 1: Overview, Relational Model / Algebra



Suyash Gupta

Assistant Professor

Distopia Labs and ONRG

Dept. of Computer Science

(E) suyash@uoregon.edu

(W) [gupta-suyash.github.io](https://github.com/suyashgupta)

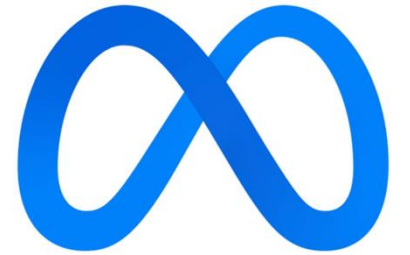
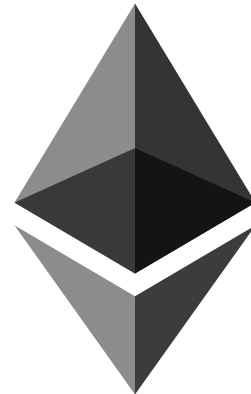


Data is Everywhere!

Data is Everywhere!



- Student Records
- Movies that you watch.
- Songs that you listen.
- Tickets that you reserve.
- Online searches.





What is a Database?



What is a Database?

- A **structured** collection of data.
- Used by all the organizations around us.



Examples of Databases

Examples of Databases



- **Colleges and Universities**

- They store your records in a database.
- These records have information about your name, age, address, grades, etc.



- **Banks**

- They store your account details in a database.
- Your account detail includes your account balance, name, age, etc.

What is a Database Management System?

What is a Database Management System?

- Also known as **DBMS**.
- Facilitates adding, storing, and retrieving data from a database.
- A software that helps manage a database.
- Examples: PostgreSQL, Oracle, IBM DB2.

Why not use Files?

- Operating System (OS) allows us to create files.
- Reading and Writing to file is possible!
- We all have created files on our laptops/PCs at some point of time.



Why not use Files?

- Operating System (OS) allows us to create files.
- Reading and Writing to file is possible!
- We all have created files on our laptops/PCs at some point of time.
- **Simply use files!**



Lets Create a File

- A file about employees at University of Oregon.
- Assume, following is the data in our file.

Name,	Age,	Job Title
Voldemort,	70,	Dark Lord
Anakin Skywalker,	85,	Darth Vader
Kang,	20,	Conqueror

cs-employees.txt

File Systems Disadvantages:

Data Redundancy and Consistency

Two different files in the system can have redundant records and/or records with inconsistent (non-identical) data.

Name,	Age,	Job Title
Voldemort,	70,	Dark Lord
Anakin Skywalker,	85,	Darth Vader
Kang,	20,	Conqueror

cs-employees.txt

Name,	Age,	Job Title
Gru,	45,	Inventor
Thanos,	100,	Eternal
Kang,	120,	TimeKeeper

math-employees.txt

For example: the entry for Kang has different age and title in the two files.

File Systems Disadvantages:

Difficulty in accessing data

- Suppose you wanted to print the name of all the CS employees with age < 80. How would you do that in this file?

Name,	Age,	Job Title
Voldemort,	70,	Dark Lord
Anakin Skywalker,	85,	Darth Vader
Kang,	20,	Conqueror

cs-employees.txt

File Systems Disadvantages:

Difficulty in accessing data

- Suppose you wanted to print the name of all the CS employees with age < 80. How would you do that in this file?
 - You can create a python or shell program that does?

Name,	Age,	Job Title
Voldemort,	70,	Dark Lord
Anakin Skywalker,	85,	Darth Vader
Kang,	20,	Conqueror

cs-employees.txt

File Systems Disadvantages:

Difficulty in accessing data

- Now, suppose you wanted to print the name of all the CS employees with age < 30 or greater than 80?

Name,	Age,	Job Title
Voldemort,	70,	Dark Lord
Anakin Skywalker,	85,	Darth Vader
Kang,	20,	Conqueror

cs-employees.txt

File Systems Disadvantages:

Difficulty in accessing data

- Now, suppose you wanted to print the name of all the CS employees with age < 30 or greater than 80?
 - So, you have to create another python or shell program?

Name,	Age,	Job Title
Voldemort,	70,	Dark Lord
Anakin Skywalker,	85,	Darth Vader
Kang,	20,	Conqueror

cs-employees.txt

For every new query, a new program! Hard to maintain.

File Systems Disadvantages:

Integrity Constraints

- Suppose you want to ensure that no employee with age > 80 should have a record in the file *cs-employees.txt*.

File Systems Disadvantages:

Integrity Constraints

- Suppose you want to ensure that no employee with age > 80 should have a record in the file *cs-employees.txt*.
- How would you ensure that if multiple people can edit the same file?

Guarantees by Database

- Data Integrity
- Data Security
- Ease of Access
- Efficient management

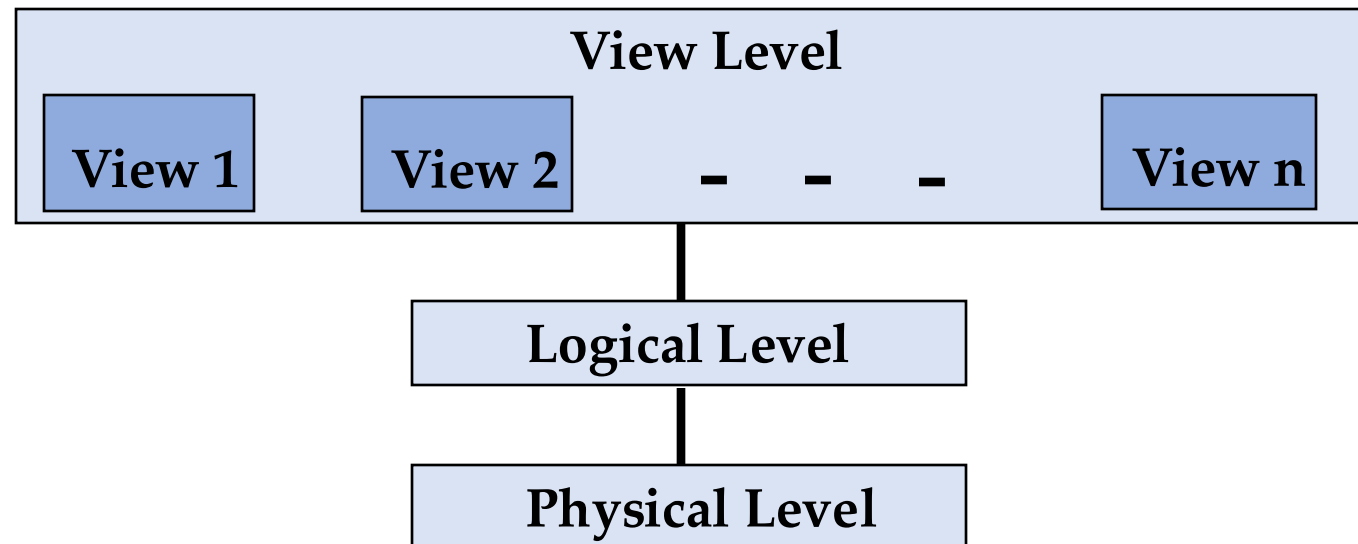


Data Abstraction

- Depending on the type of user accessing the database, we define levels of abstraction.
- How much detail is hidden or revealed at different stages.
 - Remember: All your database users are not experts!
- How many levels of abstraction?

Data Abstraction

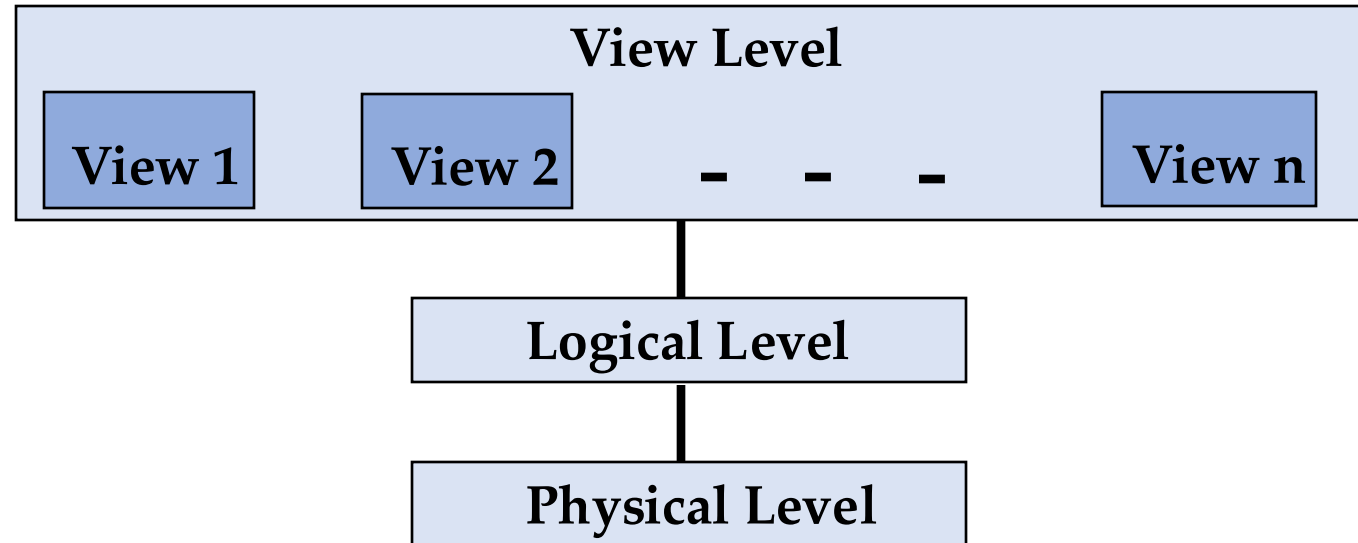
- Depending on the type of user accessing the database, we define levels of abstraction.
- How much detail is hidden or revealed at different stages.
 - Remember: All your database users are not experts!
- **Three levels:** physical, logical, and view.



Data Abstraction: Physical Level

Data Abstraction: Physical Level

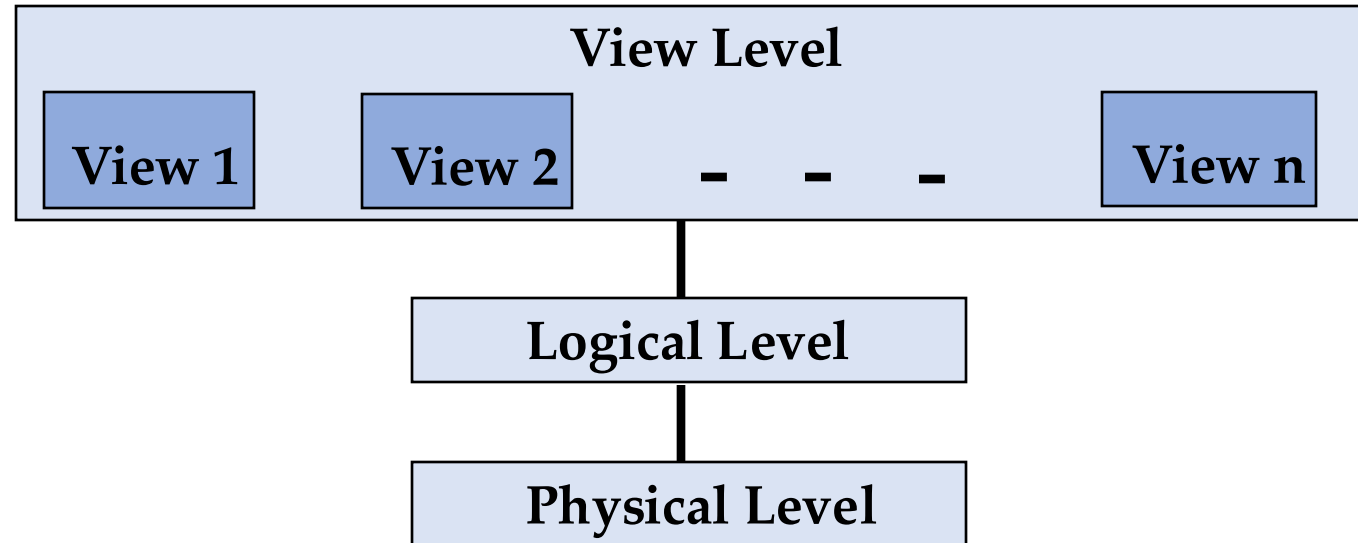
- Talks about *how* the data is stored in physical media.
- Describes data-structures that help to store the data.
- Users: Database Designers.



Data Abstraction: Logical Level

Data Abstraction: Logical Level

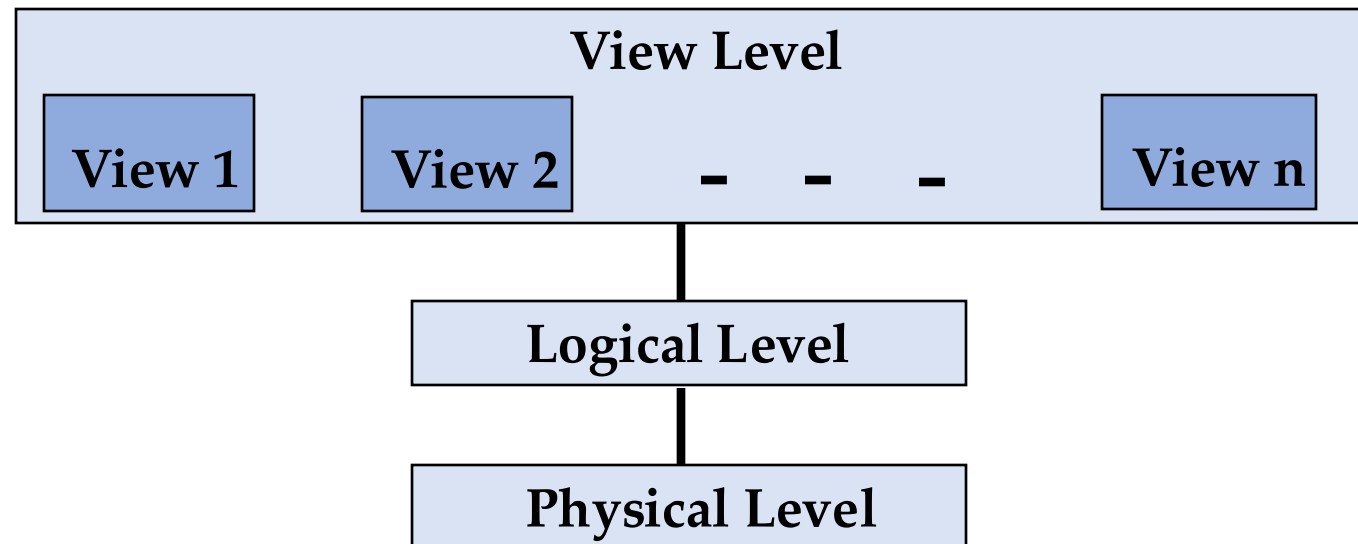
- Talks about *what* data is stored in the database (the type of data).
- Also describes the relationship between the stored data.
- Users: Database Administrators.



Data Abstraction: View Level

Data Abstraction: View Level

- Talks about parts of the data that *interests*.
- Not all the information stored in the database is of interest to every user. Thus, many views of a single database can exist.
- Users: Applications or clients.



Database Schema

Database Schema

- Schema describes the design or blueprint of the database.

Database Schema

- Schema describes the design or blueprint of the database.
- For each data abstraction level, we have a corresponding schema level:
 - **Physical schema** specifies files, storage formats, and indexes.
 - **Logical schema** lays out tables, columns, and relationships.
 - **View schema** lays out subsets or combinations of columns/tables for a particular user.
 - At view level, multiple schemas possible; often referred to as subschemas.

Physical Data Independence

- An application offers physical data independence if the modification of physical schema does not alter logical schema.
- Change in physical data-structures should not require change of data-representation at the logical level.

Schema and Abstraction

- In a University Database:
- Data abstraction defines –
 - Users see only courses and grades (view level).
 - Programmers see full table structures (logical level).
 - Engineers see block/file storage (physical level).
- Database schema implements this –
 - View schema lays out which fields are shown in each user view.
 - Logical schema defines tables/relationships.
 - Physical schema defines storage methods and indices.

Data Model

Data Model

- States the nature of a database.
- Schema can be updated—edit the file that describes the schema.
- Data model is fixed; think of it like a database software.
- Types of data-models:
 - **Relational Data Model (the focus of this course)**
 - Entity-Relationship Model
 - Object-based Data Model

Database Languages

Database Languages

- Database Languages can be divided into two categories:
- **Data-Manipulation Languages (DML)**
 - Help the user to access or manipulate the data in the database. Ex:
 - Retrieve data → also often termed as **query language** as the user queries the system for a data.
 - Insert or delete data

Database Languages

- Database Languages can be divided into two categories:
- **Data-Manipulation Languages (DML)**
 - Help the user to access or manipulate the data in the database. Ex:
 - Retrieve data → also often termed as **query language** as the user queries the system for a data.
 - Insert or delete data
- **Data-Definition Languages (DDL)**
 - Help the user to specify database schema and constraints on the data to be added to the database.
 - Ex: assertions. UO sets an assertion that any student that takes CS451 should have taken CS313.

Self-Reading Task

- Read Section 1.5 on Relational Database.
 - What is the DDL and DML in the context of Relational Database?
- How does a Relational Database differ from No-SQL database?

Relational Data-Model

Tables: Collection of Relations

- **Relational Database** → collection of relations (tables).
- **Table** → collection of relationships or tuples (rows).
- **Row** → collection of attributes (**Columns**).

Table →

Name	Age	Job Title
Voldemort	70	Dark Lord
Anakin Skywalker	85	Darth Vader
Kang	20	Conqueror
Gru	45	Inventor
Thanos	100	Eternal

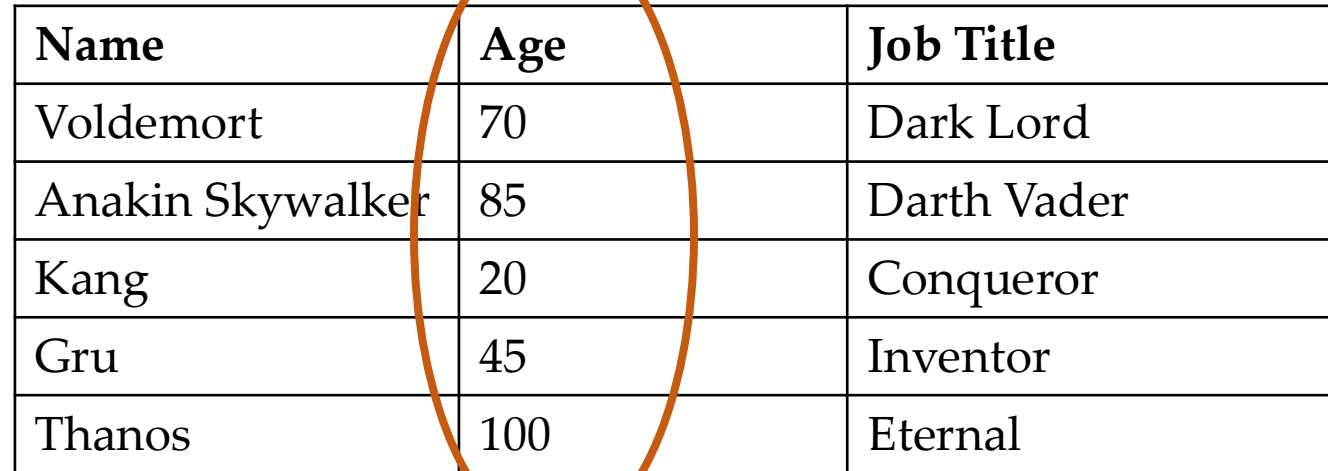
← Row

Column

Tables: Attribute Domains

Tables: Attribute Domains

- **Domain of an Attribute** → Set of permitted values.



Name	Age	Job Title
Voldemort	70	Dark Lord
Anakin Skywalker	85	Darth Vader
Kang	20	Conqueror
Gru	45	Inventor
Thanos	100	Eternal

Domain for column Age → [20,100]

Keys

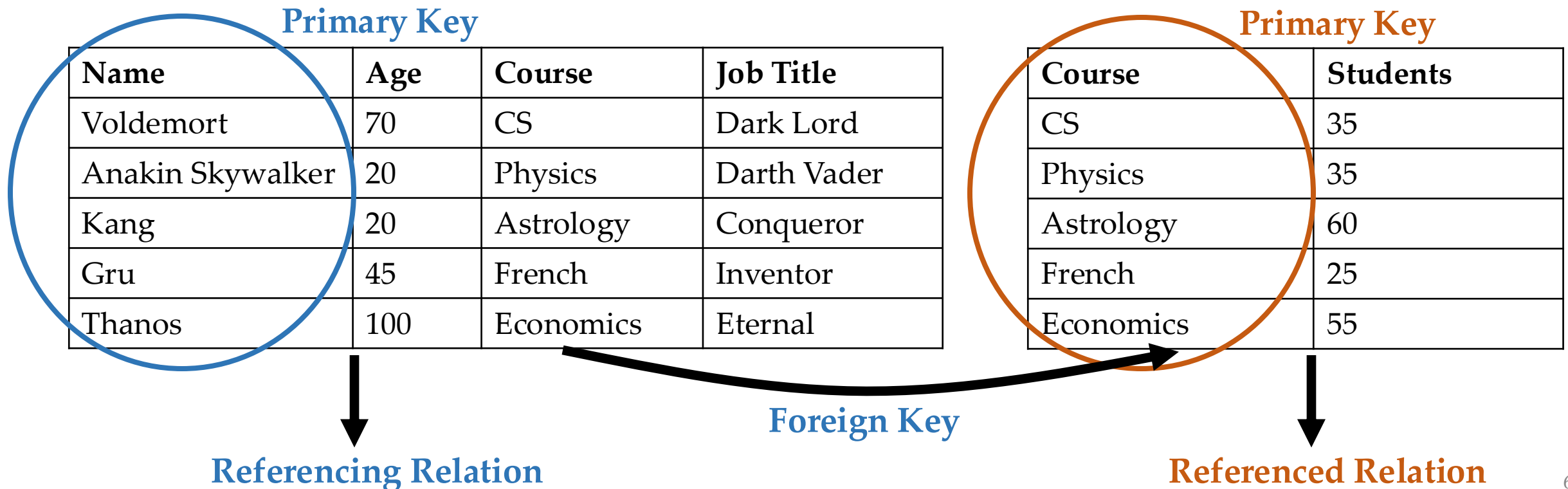
- **Superkey** → a set of one or more attributes, which together help to uniquely identify a tuple (row).
- **Candidate key** → a superkey that cannot be divided further.
- **Primary key** → a candidate key selected by the database designer as the principle way to identify tuples.

Name	Age	Job Title
Voldemort	70	Dark Lord
Anakin Skywalker	20	Darth Vader
Kang	20	Conqueror
Gru	45	Inventor
Thanos	100	Eternal

Candidate keys

Keys

- Select primary keys carefully! The value of this attribute should rarely change.
- Foreign key → a primary key of another relation (table).



Relational Algebra

Relational Algebra

- Defines a set of operations on relations.
- Allows retrieving and manipulating tuples.
- **Input** \rightarrow one or more relations.
- **Output** \rightarrow one relation.

Selection Operation (σ)

Selection Operation (σ)

- Returns rows of the relation that satisfy the predicate.
 - The predicate acts like a filter and helps to decide the necessary rows.

- Syntax: $\sigma_{\text{predicate}}(\text{relation})$

- $\sigma_{\text{Age} > 45}(\text{employees})$

employees

Name	Age	Job Title
Voldemort	70	Dark Lord
Anakin Skywalker	20	Darth Vader
Kang	20	Conqueror
Gru	45	Inventor
Thanos	100	Eternal

output

Name	Age	Job Title
Voldemort	70	Dark Lord
Thanos	100	Eternal

Projection Operation (π)

Projection Operation (π)

- Generate a relation that includes only specific attributes of rows.
 - The specific attributes are the columns of interest.
- Syntax: $\pi_{\text{attributes}}(\text{relation})$
- $\pi_{\text{Age, Name}}(\text{employees})$

employees

Name	Age	Job Title
Voldemort	70	Dark Lord
Anakin Skywalker	20	Darth Vader
Kang	20	Conqueror
Gru	45	Inventor
Thanos	100	Eternal

output

Age	Name
70	Voldemort
20	Anakin Skywalker
20	Kang
45	Gru
100	Thanos

Union Operation (U)

Union Operation (U)

- Returns a union of all the tuples in the two relations.
 - Any row that is in either of the relations is included.
- Syntax: $\text{relation}_1 \cup \text{relation}_2$
- $\text{cs-employees} \cup \text{history-employees}$

cs-employees

Name	Age
Voldemort	70
Anakin Skywalker	20

history-employees

Name	Age
Kang	20
Gru	45
Thanos	100

output

Name	Age
Voldemort	70
Anakin Skywalker	20
Kang	20
Gru	45
Thanos	100

Intersection Operation (\cap)

Intersection Operation (\cap)

- Returns an intersection of the tuples in the two relations.
 - Any row that occurs in both the relations is included.
- Syntax: $\text{relation}_1 \cap \text{relation}_2$
- $\text{cs-employees} \cap \text{history-employees}$

cs-employees

Name	Age
Voldemort	70
Anakin Skywalker	20

history-employees

Name	Age
Kang	20
Gru	45
Voldemort	70

output

Name	Age
Voldemort	70

Difference Operation (-)

- Returns a set of tuples that are present in the first relation and not the second.
- Syntax: $\text{relation}_1 - \text{relation}_2$
- $\text{cs-employees} - \text{history-employees}$

cs-employees

Name	Age
Voldemort	70
Anakin Skywalker	20

history-employees

Name	Age
Kang	20
Gru	45
Voldemort	70

output

Name	Age
Anakin Skywalker	20

Product Operation (x)

- Returns a relation that contains all possible combinations of tuples.
- Syntax: $\text{relation}_1 \times \text{relation}_2$
- $\text{cs-employees} \times \text{history-employees}$

output

Name	Age	Name	Age
Voldemort	70	Kang	20
Voldemort	70	Gru	45
Voldemort	70	Voldemort	70
Anakin	20	Kang	20
Anakin	20	Gru	45
Anakin	20	Voldemort	70

cs-employees

Name	Age
Voldemort	70
Anakin	20

history-employees

Name	Age
Kang	20
Gru	45
Voldemort	70